Instructions:

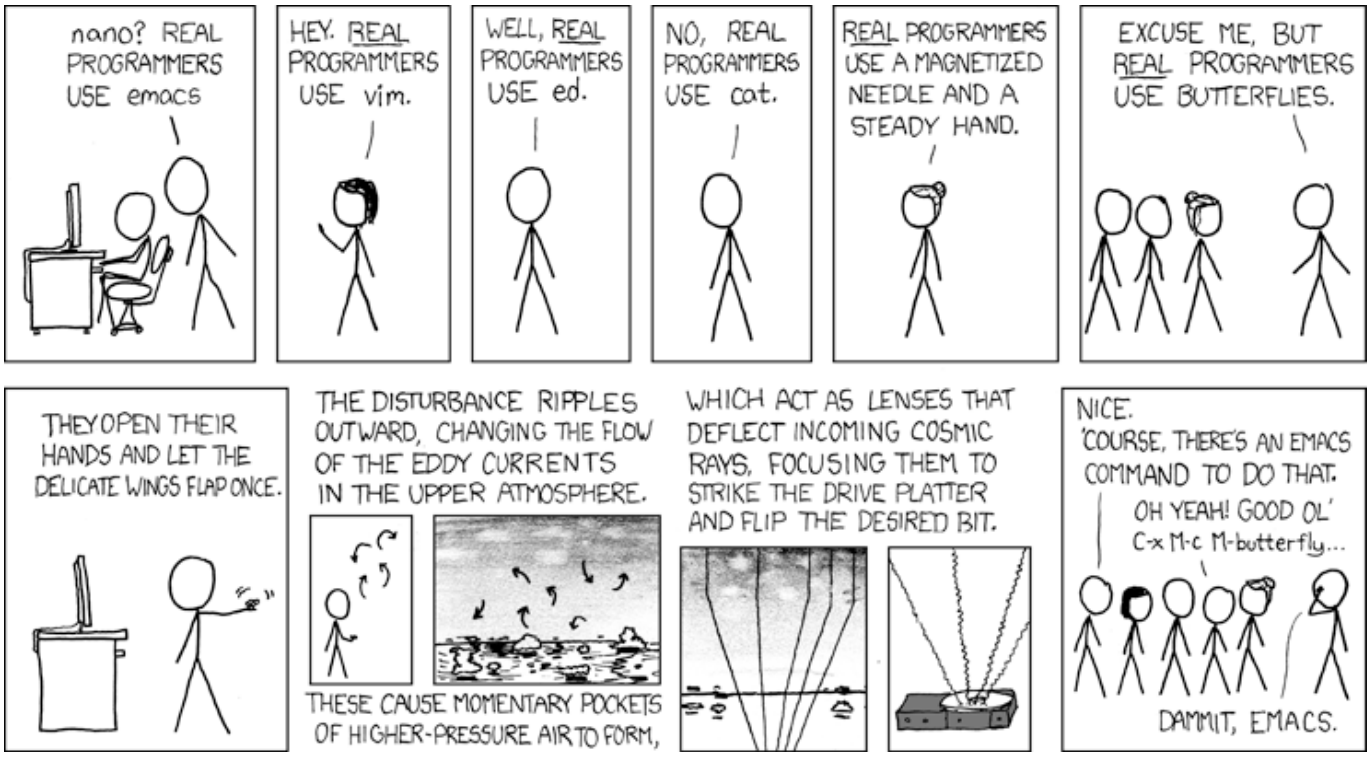
- Print your name in the space provided below.
- This examination is closed book and closed notes. No calculators or other computing devices may be used. The use of any such device will be interpreted as an indication that you are finished with the test and your test form will be collected immediately.
- Answer each question in the space provided. If you need to continue an answer onto the back of a page, clearly indicate that and label the continuation with the question number.
- If you want partial credit, justify your answers, even when justification is not explicitly required.
- There are 8 questions, some with multiple parts, priced as marked. The maximum score is 100.
- When you have completed the test, sign the pledge at the bottom of this page and turn in the test.
- Note that either failing to return this test, or discussing its content with a student who has not taken it is a violation of the Honor Code.

Do not start the test until instructed to do so!

Name ____
printed

Pledge: On my honor, I have neither given nor received unaunthorized aid on this examination.

____
signed


nano? REAL PROGRAMMERS USE emacs
HEY. REAL PROGRAMMERS USE vim.
WELL, REAL PROGRAMMERS USE ed.
NO, REAL PROGRAMMERS USE cat.
REAL PROGRAMMERS USE A MAGNETIZED NEEDLE AND A STEADY HAND.
EXCUSE ME, BUT REAL PROGRAMMERS USE BUTTERFLIES.
THEY OPEN THEIR HANDS AND LET THE DELICATE WINGS FLAP ONCE.
THE DISTURBANCE RIPPLES OUTWARD, CHANGING THE FLOW OF THE EDDY CURRENTS IN THE UPPER ATMOSPHERE.
THESE CAUSE MOMENTARY POCKETS OF HIGHER-PRESSURE AIR TO FORM,
WHICH ACT AS LENSES THAT DEFLECT INCOMING COSMIC RAYS, FOCUSING THEM TO STRIKE THE DRIVE PLATTER AND FLIP THE DESIRED BIT.
NICE.
'COURSE, THERE'S AN EMACS COMMAND TO DO THAT.
OH YEAH! GOOD OL' C-x M-c M-butterfly...
DAMMIT, EMACS.

xkcd.com

1) [10 points] True or False Questions

a) A `while` loop executes the loop body (or suite) **one or more** times.

A. True B. False

b) One reason for an infinite loop in a `while` loop is that the loop body has no statements that change the values of variables in the Boolean expression (the condition) part of the while loop.

A. True B. False

c) A variable is created when a value is assigned the first time.

A. True B. False

d) To make use the mathematical constant `math.e` you must import the `math` module.

A. True B. False

e) The = operator compares two values while the == operator assigns a value to variable.

A. True B. False

---

2) **[12 points]** Given the following declarations, evaluate the following expressions:

```
>>> a = 5
>>> b = 5
>>> c = 12
>>> d =  “Hello World!”
```

a) `a < 100 and b > 3 and b >= a` **1) true 2) false 3) Can not be determined**

b) `a == 3 or b == 3  or  a != b` **1) true 2) false 3) Can not be determined**

c) `a == 3 and c != 10` **1) true 2) false 3) Can not be determined**

d) `d != “Hello World!”` **1) true 2) false 3) Can not be determined**

---

3) **[18 points]** Give the output in the correct order that would be produced by the following statements:

```
>>> a = 5
>>> b = 2
```

a) `print(a / b)`
b) `print(a // b)`
c) `print(a + b * 4)`
d) `print(a % b)`
e) `print(13.0 / 5.0)`
f) `print("a*1.0 / b*1.0")`

OUTPUT

4) **[10 points]** Give the output that would be produced by the following code. You may assume the code executes correctly.

```
x, y = 20, 30

if (x < y):
  x = x + 10
else:
  x = x + 5

if (x == y):
  x = x + 20

else:
  x = x + 30

print(x)
```

OUTPUT

5) Consider the Python code given below. You may assume the code is syntactically correct and executes as its author intended.

```
a = int(input("Give me a number: "))
b, c = 1, 0

while b <= a:
   c = c + b
   b += 1

print("Result: ", c/(b - 1))
```

a) **[5 points]** What does the given code accomplish? Be precise, vague answers will receive little or no credit.

b) **[5 points]** Based on your answer from a) what names could you give to the variables `a`, `b`, and `c` to make the code's meaning clearer?

6) **[10 points]** Given the code below, write an `if/elif/else` statement that prints `“hello”` if `x` is **less than 30** or **greater than 60** and prints `“world”` if `x` is **between 40 and 50 inclusive**. **Any other values** should print `“foobar”`.

```
x = int(input(“Give me a number: ”))

# Your code here:
```

7) **[15 points]** Create a Python program that will sum an arbitrary number of integers entered by a user. The integers are entered one by one. The program should stop when an integer **less than or equal to zero** is entered and then print the resulting sum. Below is a sample execution where the user's input is bolded:

```
Enter number: 1
Enter number: 2
Enter number: 3
Enter number: 4
Enter number: 0
Sum is: 10
```

8) **[15 points]** Create a Python program that will compute the balance of an account that earns annually compounded interest from an initial principle and during a set period of time. The mathematical expression for annually compounded interest is $B = P * (1 + r)^t$.

`P` is the principle, `r` is the rate, and `t` is the number of years.

The user should be prompted for the principle and the number years, but you may hard code an interest rate of 5% (0.05). Below is a sample execution where the user's input is bolded:

```
Enter Principle: 1000
Enter the number of years: 4
The final balance is: 1215.51
```